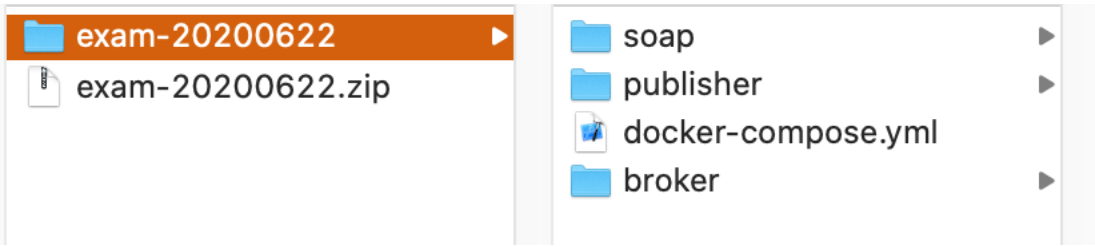
Software Engineering
MSc in Engineering in Computer Science
Sapienza Università di Roma

Programming test – June 22, 2020 – Duration 90 mins

Together with this document, a zip file has been distributed. Unzip it in a folder and you should have a structure like this:



Open the command prompt (CLI – Command Line Interface of your operating system) in the folder exam-20200622, and start the building process in Docker, i.e., run:

```
docker-compose up –build
```

At the end (it should take up to a couple of minutes, depending on your machine), three containers have been deployed and are running.

On the container *soap*, it is running a SOAP Web service *Interface* on port 8080. The WSDL is accessible at the URL: http://localhost:8080/Interface?wsdl
The service exposes functionalities about professors. In particular, the service offers the following operation (the Java signature is reported for simplicity):

- `Professor getDetails(String id)` - given an id (String) returns the full details of the given professor; a `Professor` object consists of a `String name`, a `String surname`, and a `String course` (we assume a professor teaches only one course),

On the container *broker*, on port 61616, a JMS provider offers a topic `dynamicTopics/professors` in which messages are published. The messages are published by a container *publisher.* Each message has a property `id` of type `String` (corresponding to the id of a professor whose details are offered by the Web service). The JMS provided is Apache ActiveMQ.

The student must write a client program which continuously outputs all the details of all (and only of those) professors which are mentioned in the messages (through their ids). The figure below reports an example.

```
--- exec-maven-plugin:1.2.1:exec (default-cli) @ Exam--2020-06-22--APossibleSolutionClient -
Item 10>> professor with id 4
Professor with id 4 has details Tiziana Catarci, HCI
Item 11>> professor with id 2
Professor with id 2 has details Maurizio Lenzerini, Data Management
Item 12>> professor with id 2
Professor with id 2 has details Maurizio Lenzerini, Data Management
Item 13>> professor with id 3
Professor with id 3 has details Giuseppe De Giacomo, Formal Methods
Item 14>> professor with id 4
Professor with id 4 has details Tiziana Catarci, HCI
Item 15>> professor with id 5
Professor with id 5 has details Andrea Marrella, Business Processes
Item 16>> professor with id 4
Professor with id 4 has details Tiziana Catarci, HCI
Item 17>> professor with id 5
Professor with id 5 has details Andrea Marrella, Business Processes
```

The client program can be a command line application, a Java Swing program, a servlet/jsp, whatever depending on the choice of the student. The client program should be released in the following way:

- The source code (all files in a zip file, and the pom file)
- The running jar file, with all dependencies included in it (as it has been during the labs, cf. pom files creating a single jar file)
- A txt file with the command to be executed in the CLI to run your client

The correction will consist in executing the jar file and checking that it correctly prints/shows what expected.